Project Design - LLM Generated Text Detection

Abde Manaaf Ghadiali - G29583342, Gehna Ahuja - G35741419, Sagar Sheth - G32921700, Venkatesh Shanmugam - G27887303

Experiment Objective

The objective is to train Binary Classification Models, which will ingest the data collected from the different sources mentioned below as well as in our previous documentation and classify whether the text was human-written (class 0) or AI-generated (class 1). Later, we'll perform ensemble techniques on top of those models, to give our result. Finally, we shall compare the output metrics of each of our approaches and come to a conclusion of which approach works the best, why is it better than the others and how does it classify a given text as AI generated using Explainable Techniques.

Methodology

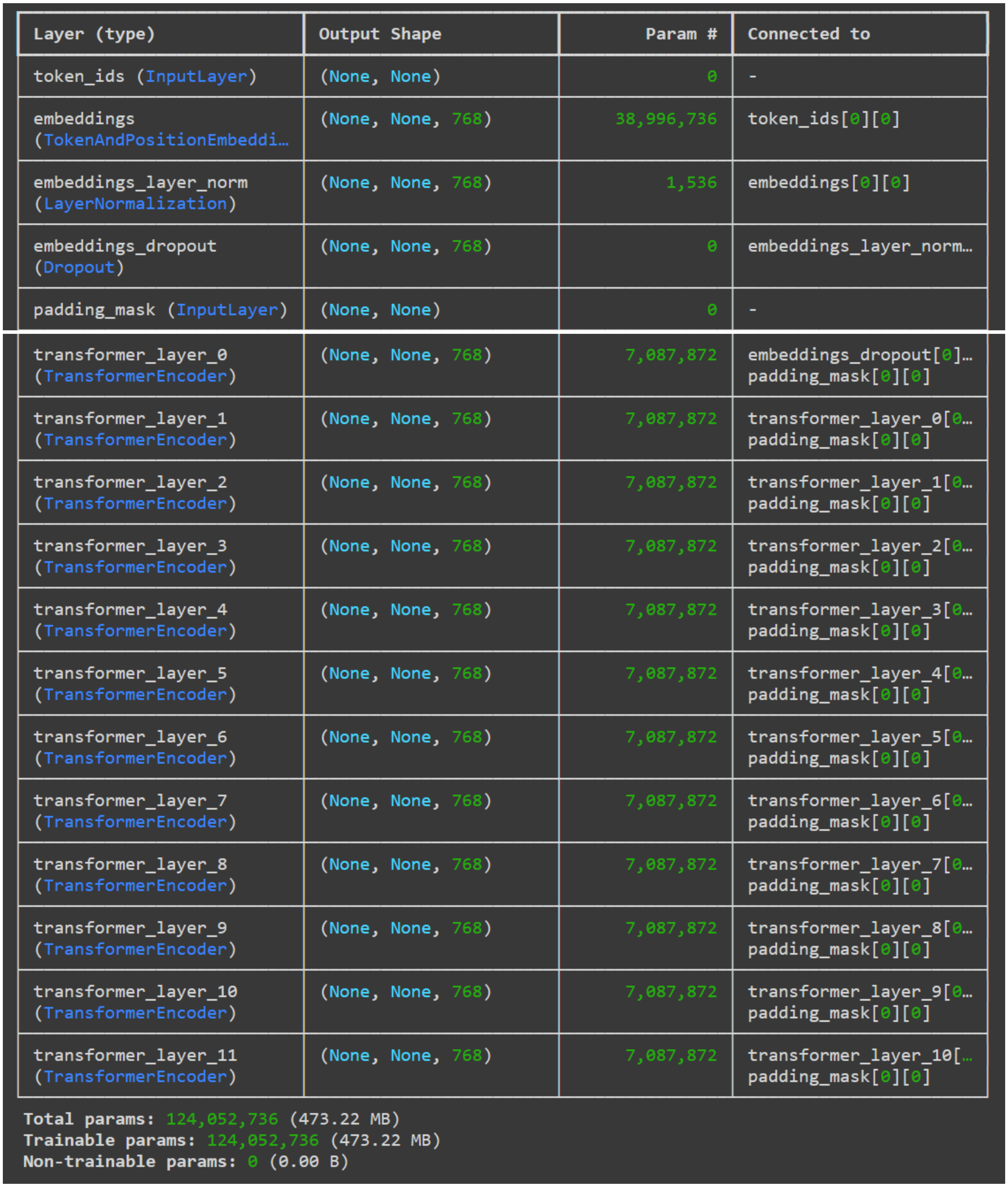
Having the AI and Human generated text we are going to perform some analysis on the data like number of stop words used by the LLM and Humans, mistakes made by them, usage of punctuation marks and so on. This Analysis will be useful to either accept or discard some hypothesis on the distinction between AI generated text from Human Written text.

Once we are done with that, we are going to extract the features out by doing TF-IDF, collocations, and word embeddings. After performing all these analysis and feature extraction techniques on our data, we will be attempting two approaches to building the Binary Classification models. We will be using transfer learning to train our data on existing models used in prior work and build classical Deep Neural Networks as baselines to compare performance.

Transfer Learning Approach:

Transfer learning architectures for classifying AI-generated text from human-written text leverage pre-trained language models, such as BERT or GPT, as foundational knowledge. These architectures fine-tune the pre-trained models on labeled datasets specific to the classification task, allowing them to adapt and specialize in distinguishing between AI-generated and human-written text. By transferring knowledge from broad language understanding tasks to the targeted classification problem, these architectures enable efficient and accurate identification of nuanced differences in language usage and coherence.

- **RoBERTa-Base [6]:** Roberta Base excels in distinguishing between AI-generated and human-written text. Trained on vast datasets, it comprehensively learns the patterns of human language, enabling it to distinguish subtle differences between AI-generated and authentic content. Leveraging its bidirectional transformers, Roberta Base efficiently identifies patterns and inconsistencies indicative of AI text generation.



```
┌───────────────────────────┬────────────────────────┬────────────────┬────────────────────────┐
│ Layer (type)              │ Output Shape           │        Param # │ Connected to           │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ token_ids (InputLayer)    │ (None, None)           │              0 │ -                      │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ embeddings                │ (None, None, 768)      │     38,996,736 │ token_ids[0][0]        │
│ (TokenAndPositionEmbeddi… │                        │                │                        │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ embeddings_layer_norm     │ (None, None, 768)      │          1,536 │ embeddings[0][0]       │
│ (LayerNormalization)      │                        │                │                        │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ embeddings_dropout        │ (None, None, 768)      │              0 │ embeddings_layer_norm… │
│ (Dropout)                 │                        │                │                        │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ padding_mask (InputLayer) │ (None, None)           │              0 │ -                      │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_0       │ (None, None, 768)      │      7,087,872 │ embeddings_dropout[0]… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_1       │ (None, None, 768)      │      7,087,872 │ transformer_layer_0[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_2       │ (None, None, 768)      │      7,087,872 │ transformer_layer_1[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_3       │ (None, None, 768)      │      7,087,872 │ transformer_layer_2[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_4       │ (None, None, 768)      │      7,087,872 │ transformer_layer_3[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_5       │ (None, None, 768)      │      7,087,872 │ transformer_layer_4[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_6       │ (None, None, 768)      │      7,087,872 │ transformer_layer_5[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_7       │ (None, None, 768)      │      7,087,872 │ transformer_layer_6[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_8       │ (None, None, 768)      │      7,087,872 │ transformer_layer_7[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_9       │ (None, None, 768)      │      7,087,872 │ transformer_layer_8[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_10      │ (None, None, 768)      │      7,087,872 │ transformer_layer_9[0… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
├───────────────────────────┼────────────────────────┼────────────────┼────────────────────────┤
│ transformer_layer_11      │ (None, None, 768)      │      7,087,872 │ transformer_layer_10[… │
│ (TransformerEncoder)      │                        │                │ padding_mask[0][0]     │
└───────────────────────────┴────────────────────────┴────────────────┴────────────────────────┘
 Total params: 124,052,736 (473.22 MB)
 Trainable params: 124,052,736 (473.22 MB)
 Non-trainable params: 0 (0.00 B)
```

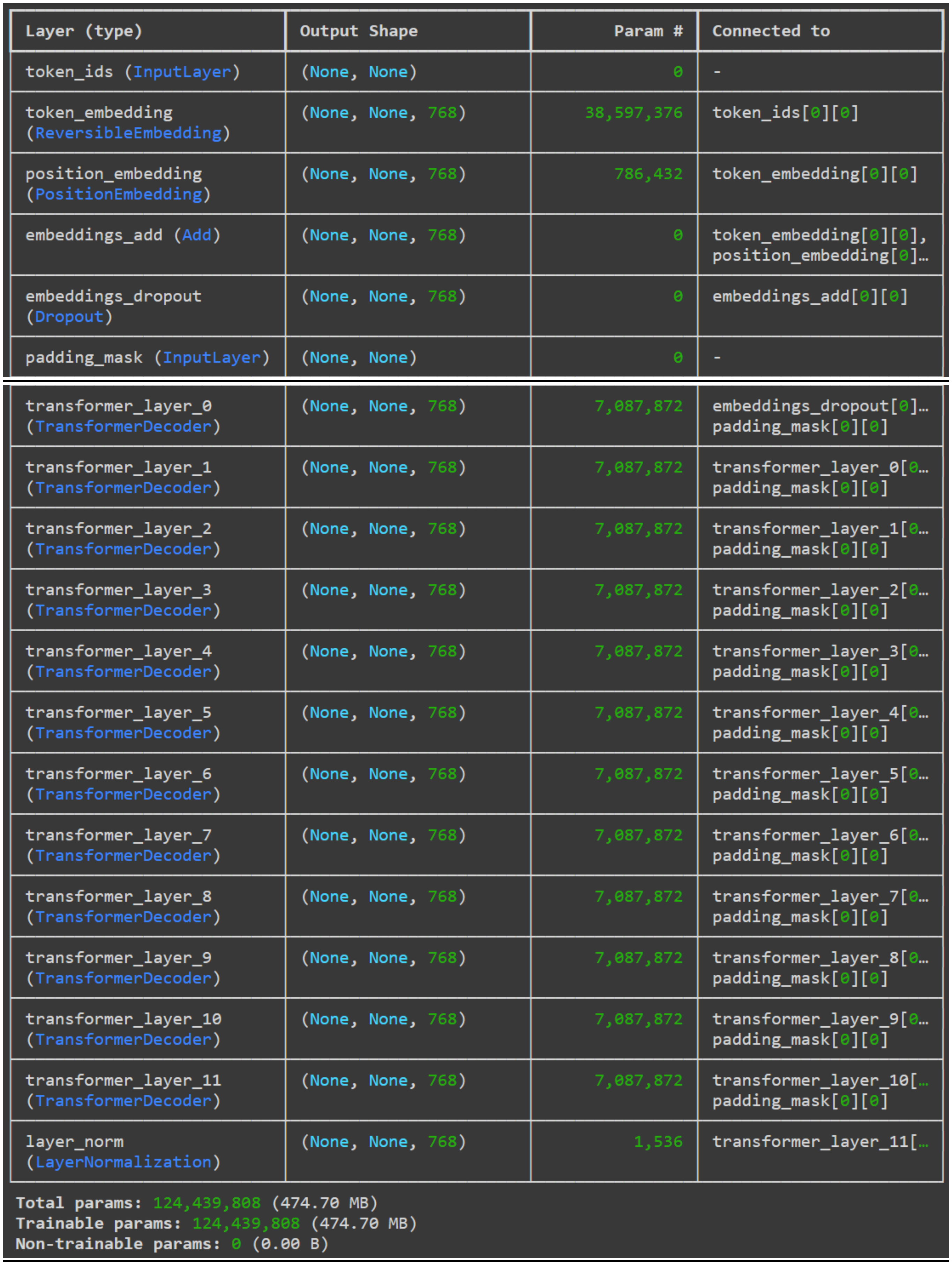
- **RoBERTa-Large [6]: B**uilding on the capabilities of Roberta Base, the Roberta Large model extends its prowess in classifying AI-generated text from human-written content. With a staggering 355 million parameters, Roberta Large delves deeper into linguistic intricacies, enhancing its ability to distinguish subtle discrepancies indicative of machine-generated text. Leveraging its extensive training on vast datasets, it employs bidirectional transformers to thoroughly analyze context and syntactic structures.

  The Architecture of the Roberta Large Model is like the Roberta Base Model. The major differences are that Roberta Large has more transformer layers (Roberta Base has 11 while Roberta Large has 23) and that it is built on more parameters.

- **LLaMA [7]:** Llama employs an Input Encoding layer to tokenize input text, converting it into numerical representations via embedding. These embeddings encapsulate semantic nuances of words. Subsequently, the encoded embeddings traverse multiple transformer layers, comprising self-attention mechanisms and feed-forward neural networks. Self-attention facilitates capturing interdependencies among input sequence words, while feed-forward networks extract task-relevant features. Post-transformer layers, representations of each token are aggregated, yielding a fixed-size sequence representation (vector), achievable through global average pooling or max pooling. Logits undergo SoftMax activation, transforming them into probabilities, where the class with the highest probability signifies the predicted class.

  Llama's distinguishing feature lies in its architecture, which integrates specific mechanisms tailored for semantic understanding and feature extraction, potentially setting it apart from Roberta in terms of efficiency and performance in classifying AI-generated and human-written text.

- **GPT-2 [8]:** The GPT-2 model features a transformer architecture with multi-head self-attention mechanisms, facilitating comprehensive contextual understanding of input text. It consists of numerous layers, each refining representations through iterative processing. During inference, the model predicts the likelihood of the next word given the preceding context, enabling seamless generation of coherent text. To classify AI-generated from human-written text, fine-tuning techniques are applied. By training on labeled datasets, GPT-2 learns to discern subtle differences in language patterns and semantic coherence. Its robust architecture and adaptability empower it to effectively classify text, contributing to efforts in combating misinformation and ensuring content authenticity.



```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                ┃ Output Shape             ┃         Param # ┃ Connected to               ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ token_ids (InputLayer)      │ (None, None)             │               0 │ -                          │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ token_embedding             │ (None, None, 768)        │      38,597,376 │ token_ids[0][0]            │
│ (ReversibleEmbedding)       │                          │                 │                            │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ position_embedding          │ (None, None, 768)        │         786,432 │ token_embedding[0][0]      │
│ (PositionEmbedding)         │                          │                 │                            │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ embeddings_add (Add)        │ (None, None, 768)        │               0 │ token_embedding[0][0],     │
│                             │                          │                 │ position_embedding[0]…     │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ embeddings_dropout          │ (None, None, 768)        │               0 │ embeddings_add[0][0]       │
│ (Dropout)                   │                          │                 │                            │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ padding_mask (InputLayer)   │ (None, None)             │               0 │ -                          │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_0         │ (None, None, 768)        │       7,087,872 │ embeddings_dropout[0]…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_1         │ (None, None, 768)        │       7,087,872 │ transformer_layer_0[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_2         │ (None, None, 768)        │       7,087,872 │ transformer_layer_1[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_3         │ (None, None, 768)        │       7,087,872 │ transformer_layer_2[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_4         │ (None, None, 768)        │       7,087,872 │ transformer_layer_3[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_5         │ (None, None, 768)        │       7,087,872 │ transformer_layer_4[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_6         │ (None, None, 768)        │       7,087,872 │ transformer_layer_5[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_7         │ (None, None, 768)        │       7,087,872 │ transformer_layer_6[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_8         │ (None, None, 768)        │       7,087,872 │ transformer_layer_7[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_9         │ (None, None, 768)        │       7,087,872 │ transformer_layer_8[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_10        │ (None, None, 768)        │       7,087,872 │ transformer_layer_9[0…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ transformer_layer_11        │ (None, None, 768)        │       7,087,872 │ transformer_layer_10[…     │
│ (TransformerDecoder)        │                          │                 │ padding_mask[0][0]         │
├─────────────────────────────┼──────────────────────────┼─────────────────┼────────────────────────────┤
│ layer_norm                  │ (None, None, 768)        │           1,536 │ transformer_layer_11[…     │
│ (LayerNormalization)        │                          │                 │                            │
└─────────────────────────────┴──────────────────────────┴─────────────────┴────────────────────────────┘
 Total params: 124,439,808 (474.70 MB)
 Trainable params: 124,439,808 (474.70 MB)
 Non-trainable params: 0 (0.00 B)
```

- **GPT-3 [9]:** GPT-3, an evolution of its predecessor GPT-2, boasts a larger architecture with 175 billion parameters, enabling deeper contextual understanding and more nuanced language generation. Its transformer-based architecture encompasses numerous layers of multi-head self-attention mechanisms, facilitating intricate linguistic analysis and text generation.

  Building on GPT-2's capabilities, GPT-3 exhibits enhanced adaptability and proficiency in discerning between AI-generated and human-written text. By fine-tuning labeled datasets and leveraging its expansive knowledge base, GPT-3 refines its ability to classify text with greater accuracy.

Deep Neural Networks from Ground Up:
Deep Neural Networks architectures for classifying AI-generated text from human-written text involve designing neural network architectures specifically tailored for the task. These architectures typically consist of multiple layers of neurons, with each layer responsible for extracting and learning different features from the input text data. By building the network from scratch, these architectures offer flexibility in customization and optimization for the classification task, enabling effective capture of subtle linguistic cues and patterns inherent in AI-generated and human-written text.

- **Convolutional Neural Network (CNN):** We will utilize one-dimensional convolutional layers. These convolutional layers will apply filters over the input text and capture local patterns or features. Pooling layers (either max pooling or average pooling) will down sample the feature maps, retaining important information while reducing dimensionality. Fully connected layers at the end of the network will integrate the learned features for classification.

- **Recurrent Neural Network (RNN):** RNNs process sequential input text word by word, utilizing recurrent connections to maintain a hidden state capturing context. Each word will be sequentially fed into the network, updating the hidden state based on previous words. The final hidden state will encode the entire sequence, which will then be used for classification through fully connected layers.

- **Long Short-Term Memory (LSTM):** LSTMs are a type of RNN specifically designed to address the vanishing gradient problem and capture long-range dependencies. LSTMs process input sequences similarly to traditional RNNs but with the advantage of better preserving information over long distances. Like RNNs, LSTMs typically use a final hidden state for classification, which we may modify with additional layers for improved performance.

After training the data on various models such as Roberta, or GPT, each model will generate predictions for the given classification task. The next step involves selecting the top-performing models based on their individual prediction accuracies and other evaluation metrics. These top 3 to 5 models are then combined using ensemble techniques to produce the final prediction.

Ensembling techniques can include methods like majority voting, where the final prediction is determined based on the most common prediction among the ensemble of models, or weighted averaging, where predictions from each model are weighted based on their performance. Other methods such as stacking or boosting may also be employed, where models are combined in more sophisticated ways to leverage their individual strengths.

Evaluation

The baseline performance will involve classifying the input text at random, yielding an accuracy of 0.5. Achieving results beyond this threshold will be considered a success. The primary metric for the scope of this project will be the F1 Score, as both False Negatives and False Positives are equally important. Additionally, the correct context for explaining why our model is classifying a given text as AI-generated would be a critical factor for success.

We assume compared to previous work our results the performance of our model would be similar or marginally better given that we are attempting to ensemble our results over multiple model architectures.

Data Sources and Information

The data we will be using is a corpus that contains both Human Generated & AI Generated Data. As we have various datasets with varying columns as initial steps, we will take only those columns into account which are required and drop the others to make our data even. Once we have done that, we are going to combine all the datasets into one.

We will use this data and extract features like how often the LLM’s use stop words compared to humans, checking for mistakes as we assume that LLM’s don’t make mistakes. After performing some EDA on a small amount of our data it gives confidence that the data is enough as a starting point for creating a binary classifier that accurately distinguishes between LLM vs Human generated text.

Data	Source	Data Format	Rows	Additional Information
Human vs LLM Text Corpus | Kaggle	Kaggle	.csv .parquet	786k	**text**: Contains the Human Generated and the ML Generated text **source**: How the text was generated **generated**: 0/1 Indicating if the Text is Human/ML Generated **text_length:** Number of Characters in the text **word_count**: Number of words in the text
DAIGT v3 Train Data (Human and AI) | Kaggle	Kaggle	.csv	65k	**text:** Contains the Human Generated and the ML Generated text **label:** 0/1 Indicating if the Text is Human/ML Generated **prompt name:** prompt **source** **RDizzl3_seven:** N/a
LLM - Detect AI Generated Text | Kaggle	Kaggle	.csv	1378	**text**: Contains the Human Generated and the ML Generated text **generated**: 0/1 Indicating if the Text is Human/ML Generated
Deepfake Text Detect | GitHub	GitHub	.csv	445k	**text:** Contains the Human Generated and the ML Generated text **label:** 0/1 Indicating if the Text is Human/ML Generated **source**: How the text was generated
Essay With Instructions | HuggingFace	Hugging Face	.csv	3.4k	**Txt:** Contains the Human Generated **label:** 0 Indicating the human written text **Instruction:** Topic of the text written

References

[1] G. Gritsay, A. Grabovoy and Y. Chekhovich, "Automatic Detection of Machine Generated Texts: Need More Tokens," 2022 Ivannikov Memorial Workshop (IVMEM), Moscow, Russian Federation, 2022, pp. 20-26, doi: 10.1109/IVMEM57067.2022.9983964.

[2] Gaggar, Raghav, Ashish Bhagchandani, and Harsh Oza. "Machine-Generated Text Detection using Deep Learning." arXiv preprint arXiv:2311.15425 (2023).

[3] Jawahar, Ganesh, Muhammad Abdul-Mageed, and Laks VS Lakshmanan. "Automatic detection of machine generated text: A critical survey." arXiv preprint arXiv:2011.01314 (2020).

[4] Kadhim Hayawi, Sakib Shahriar, Sujith Samuel Mathew," The Imitation Game: Detecting Human and AI-Generated Texts in the Era of ChatGPT and BARD",doi.org/10.1177/016555152412275.

[5] Wenxiong Liao, Zhengliang Liu, Haixing Dai, Shaochen Xu, Zihao Wu, Yiyang Zhang, Xiaoke Huang, Dajiang Zhu, Hongmin Cai, Tianming Liu, Xiang Li, "Differentiating ChatGPT-Generated and Human-Written Medical Texts: Quantitative Study", doi: 10.2196/48904

[6] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

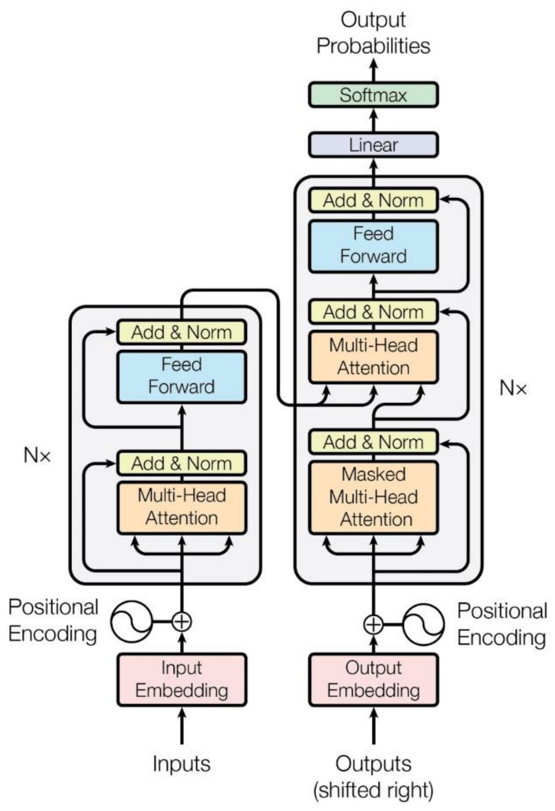
[7] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.

[8] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

[9] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

Appendix

The transformer architecture, notably used in models like BERT and GPT, revolutionizes text classification tasks. It employs self-attention mechanisms to capture contextual dependencies between words in both AI-generated and human-written text. By analyzing the entire input sequence simultaneously, transformers excel at understanding long-range dependencies and semantic relationships within the text.


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)